CAMPUS
DE EXCELENCIA
INTERNACIONAL

Graduado en Ingeniería Informática

Universidad Politécnica de Madrid

Facultad de Informática

TRABAJO FIN DE GRADO

Desarrollo aplicación web y móvil para reservas en comedores

Autor: Antonio Blázquez Bea

Director: M. Luisa Cordoba Cabeza

MADRID, JUNIO DE 2012

ÍNDICE

1. RESUMEN

Este trabajo trata sobre el desarrollo de una “Start Up” de base tecnológica desde la universidad. La empresa creada comercializará un servicio software para comedores y restaurantes.

La actualidad está marcada por la economía, y en estos momentos nos encontramos con una tasa de paro general del 26,7%, en ingenierías (la UPM) es del 31,2%. El sector de las TIC se salva por el momento (en informática la tasa de paro es del 4,8%), aunque el trabajo en muchos casos es precario y las oportunidades para desarrollar una carrera plena pasan por ir a trabajar a otros países que tienen una situación laboral más favorable. También tenemos la alternativa de trabajar como autónomos, creando empresas y, con suerte, generando un nuevo empleo.

En España existen iniciativas de ayuda pública y privada que ayudan a convertir ideas en empresas, incubadoras, aceleradoras, ayudas públicas, ferias tecnológicas, concursos que ayudan a ganar visibilidad, etc. Este proyecto nació gracias a una de estas iniciativas, más en concreto a ACTUA UPM, iniciativa de la propia UPM a través de la unidad Creación de Empresas.

Para crear una empresa no basta un buen servicio o un buen producto, es necesario poder venderlo. Por lo que el primer esfuerzo debe realizarse en comprobar que nuestro producto tiene características que pueden cubrir una necesidad que ya existe y que las condiciones indican que puede ser rentable. Esta es la primera fase, la creación de un plan de negocio y de marketing.

A continuación basándonos en la manera en la que decidimos que venderemos nuestro producto, orientaremos el desarrollo para hacer hincapié en los puntos fuertes que pueden hacer nuestro producto diferente y deseado.

La fase de desarrollo es mi caso es sobre todo una fase de aprendizaje, en la que aprenderé a fondo tecnologías web y móvil, y aplicaré los conocimientos adquiridos en mis estudios.

A continuación, se ofrece mi experiencia desarrollando la empresa desde la idea hasta conseguir un producto preparado para ponerlo a prueba.

ABSTRACT

This paper is about the development of a technological based “Start Up” from the college. The company will market a software service oriented to canteens and restaurants.

Economy is what rules the world, and today we are facing a very strong crisis with a 26.7% of unemployment rate in Spain. However, the IT sector is less affected than others by this crisis (Informatics Engineers has a 4.8% unemployment rate). But in many cases jobs are precarious and young people have to leave our country for pursuing a decent career. Also we have the option of self-employment, launching a company, and with some luck, creating a new job.

We have some tools for launching new technological base business. Many people hopes to create the new Facebook, and many investors are interested in being on the boat if that happens. Also at UPM we have ACTUA UPM, which promotes ideas into companies. My idea was born in it and managed to the final round.

For launching a new company you need to do a business plan that studies the possible pitfalls of your idea and directs your development efforts to the way your product is going to be sold. So the first phase of this paper will talk about the development of the business plan.

After it, the development phase is in essence a phase of learning in which I faced most problems in my own, giving the best solutions I could from my own experience and intuition.

Then, we present the experience of developing a Start Up from the idea to the market testing.

2. INTRODUCCIÓN Y OBJETIVOS

2.1 JUSTIFICACIÓN

Los informáticos tenemos la suerte de poder desarrollar productos con sólo una idea y un ordenador portátil. Fue lo que me llevó a elegir esta carrera y este trabajo es el primer fruto de mi elección. La idea de crear una empresa surgió a razón de la necesidad que tuve de dar solución a un problema práctico con una herramienta informática.

En los comedores a los que acudo diariamente, tanto en la universidad como en la residencia universitaria en la que he estado, ofrecen menús variados todos los días. En estos comedores siempre sobra comida, básicamente porque ofrecen varios platos para elegir y tienen que calcular siempre un extra basándose en la experiencia que tengan de anteriores días y hay días que por razones externas faltan muchos clientes (exámenes, eventos deportivos, etc.). Éste hecho es ampliable a otros comedores.

Con el acceso global a internet es posible implementar una herramienta que permita organizar un sistema de reservas de comidas útil para comedores a los que prácticamente siempre acuden los mismos clientes, como pueden ser los comedores escolares, universitarios, residenciales, o los que surgen alrededor de los núcleos empresariales. Si un comedor es capaz de prever con relativa exactitud la cantidad de comida que necesita, también reducirá sus costes y aumentará sus beneficios.

Las personas que se benefician de estos servicios online son una excelente base de clientes para dar un servicio de comidas a domicilio, que puede ser un servicio futuro que ofrezca la empresa si se alcanza la masa crítica necesaria. Las personas que acuden a comedores normalmente no cocinan en casa y gastan más en comer en restaurantes. Si a esto le sumamos que utilizarán la página diariamente para reservar sus menús diarios, tenemos a un cliente con confianza en el producto y con alta probabilidad de realizar un pedido de comidas a domicilio.

2.2 OBJETIVOS

Por lo antes descrito, los objetivos propuestos para desarrollar la empresa han sido:

1. Realizar un Plan de Negocios para el desarrollo de una aplicación que ofrezca a los restauradores un producto útil en su día a día y que les reporte beneficios, además de un producto para los usuarios que les permita ojear, reservar y comprar los menús de sus restaurantes habituales.
2. Estimación y diseño del proyecto para tener una previsión de esfuerzo, costes y planificación que se requerirán para realizar todas las actividades e implementar

las plataformas web y móvil asociadas con el trabajo. Aplicar los conocimientos de ingeniería del software para la elaboración de la arquitectura de la aplicación.

3. Desarrollar el proyecto, desde el diseño de la interfaz, la interacción del usuario y la codificación de la aplicación.

3. DESARROLLO

En los siguientes puntos mostraré los datos recogidos de la realización de las diferentes tareas y resumiré en que ha consistido cada una de ellas.

3.1 Plan de negocio

Un plan de negocio es una declaración formal de un conjunto de objetivos de una idea o iniciativa empresarial, que se constituye como una fase de proyección y evaluación. Se emplea internamente por la administración para la planificación de la empresa y complementariamente, es útil para convencer a terceros, tales como bancos o posibles inversores (p. ej. los business angels o las empresas de capital riesgo), para que aporten financiación al negocio.

En resumidas cuentas cada tarea que aquí expongo es un punto fundamental que todo plan de negocio debe contener.

De cada tarea realizada aportaré una descripción y una las conclusiones finales que he sacado después de realizarla, problemas encontrados y soluciones elegidas.

3.1.1 Descripción del proyecto

Descripción: Se trata de resumir la idea de negocio y realizar un análisis resumido de viabilidad. Es la parte más importante de un plan de negocio porque es la primera que los inversores van a leer y dependiendo del interés generado en éste punto seguirán leyendo o no.

Postmortem: Es importante realizarlo al concluir la redacción del plan de negocio para poder transmitir la idea global en un par de páginas.

3.1.2 Análisis del sector de inversión

Descripción: Desde un punto de vista económico se realiza un informe de la situación del país, del sector y de la dinámica del sector para obtener datos reales con los que sustentar los siguientes puntos del informe.

Postmortem: Falta de comprensión económica avanzada. Las personas que van a leer el informe saben mejor que tú la situación del sector (si realmente les interesa).

3.1.3 Análisis e investigación de mercado

Descripción: La investigación de mercado se utiliza para conocer la oferta, es decir cuáles son las organizaciones o negocios similares y qué beneficios ofrecen y, asimismo, para conocer la demanda, es decir quiénes son y qué quieren los consumidores. Incluye estudio de mercado, entrevistas con futuros clientes, análisis de la competencia y de la demanda, entre otras.

Postmortem: En mi opinión es el punto más importante de un plan de negocio. Te permite ver donde encaja tu producto.

3.1.4 Análisis externo

Descripción: Todas las empresas se ven afectadas por los factores de su entono, aunque no sea con la misma intensidad en todos los casos, y es conveniente realizar un análisis detallado de este entorno para poder identificar posibles oportunidades o amenazas que puedan surgir del mismo.

Postmortem: Es un punto muy subjetivo, ya que depende completamente de la visión propia del entorno. Puede que nos equivoquemos al definir un competidor o que no veamos alguno.

3.1.5 Análisis interno

Descripción: Un análisis de los diferentes factores que puedan existir dentro de una empresa, permite evaluar los recursos con que cuenta la empresa para conocer su situación y capacidades, y asimismo detectar fortalezas y debilidades, con el fin de diseñar estrategias que permitan potenciar o aprovechar las fortalezas, y neutralizar o eliminar las debilidades.

Postmortem: Realmente te hace pensar que tipo de empresa eres y que estás dispuesto a hacer para conseguir llegar a donde quieres llegar.

3.1.6 Análisis de las oportunidades de negocio

Descripción: Este análisis es un método sencillo y eficaz que permite identificar los factores estratégicos críticos de un negocio, para plantear las acciones que se deben poner en marcha, consolidando las fortalezas y aprovechando las ventajas de las oportunidades detectadas, con el fin de apoyar el éxito del negocio.

Postmortem: Este punto refuerza los anteriores. Resume las fortalezas de tu negocio, y cuáles son las novedades que trae consigo al sector.

3.1.7 Plan de marketing

Descripción: Es un análisis de las características que se pueden publicitar de tu negocio y permite centrarse en desarrollar un producto acorde a lo que queremos vender a los clientes siempre y cuando cumpla con los objetivos.

Postmortem: Es complicado hacer un plan de marketing de algo que no existe, pero por otro lado ayuda a identificar los aspectos del producto que deben cuidarse más durante su desarrollo.

3.1.8 Aspectos legales

Descripción: Los requisitos legales para tiendas online y la venta por internet son los mismos que para cualquier otro tipo de negocio. En este caso son los mismos requisitos que para una tienda física de la calle, pero con la diferencia que las tiendas online no tienen que cumplir los trámites de licencias de apertura.

Postmortem: Información muy útil que nunca está de más conocer.

3.2 Estimación del proyecto

Predicción de esfuerzo, costes y planificación que se requerirá para realizar todas las actividades y construir todos los productos asociados con el proyecto.

Existen multitud de métricas que se pueden aplicar. Para mi proyecto las que he considerado más relevantes han sido las de LOC (líneas de código) y puntos de función. Esto permite medir el tamaño por un lado por funcionalidad y por líneas de código y por otro lado nos permite estimar la dificultad de cada función que realicemos.

Lo normal al principio es quedarse corto debido a que no valoramos la complejidad de funcionalidades que no hemos desarrollado nunca.

En el desarrollo de este proyecto he subestimado el tiempo que iba a tener que dedicar al diseño de la interfaz. Aunque había realizado páginas web en el pasado nunca había tenido que realizar un diseño robusto para diferentes plataformas, lo que hizo que tuviera que rehacer el diseño hasta en 4 ocasiones.

A continuación las tareas realizadas.

3.2.1 Requisitos de la aplicación

Descripción: Se trata de poner por escrito todos los servicios que los usuarios de la aplicación van a recibir. El modo en el que quiero que los reciban y cómo debe interactuar el usuario con el sistema.

Postmortem: Pensar la mejor forma de realizar una acción a la primera es casi imposible. Los requisitos han ido cambiando a medida que la aplicación tomaba forma.

3.2.2 Cálculo de los puntos de función

Descripción: Se trata de valorar el esfuerzo para desarrollar las funcionalidades del sistema, la dificultad que tiene y de agrupar los requisitos en funcionalidades.

Postmortem: La primera estimación fue mala, porque se tuvieron que rehacer funcionalidades completas después de cambiar el diseño de la interfaz.

3.3 Diseño del proyecto

Descripción: Antes de comenzar a programar la aplicación es necesario desarrollar el diseño sobre papel para tener una referencia clara desde el comienzo. Seguramente será necesario hacer modificaciones en el futuro, pero ante proyectos complejos este paso es completamente necesario y fundamental.

Si en el futuro necesitamos realizar modificaciones en una parte del proyecto, de un vistazo a los diagramas de clases, de secuencia y al de la base de datos podremos saber todas las dependencias que existen con otros módulos.

Además si en el futuro alguien se incorpora al desarrollo del proyecto es fundamental contar con diagramas para que pueda asimilar la estructura de la aplicación lo antes posible.

Postmortem: He realizado modificaciones en el diseño debido a que los requisitos cambiaron. La manera de interactuar del usuario también fue repensada, sobre todo porque a la hora del diseño de la interfaz, te pones en la piel del usuario y con suerte llegas a comprender como pensará al utilizar tu sistema.

3.3.1 Diagramas de flujo de datos (DFD's)

Descripción: Después de haber identificado cada objeto del sistema (por ejemplo Usuarios, Comedores, etc…), se trata de definir los acontecimientos que provocan la entrada o salida de información de estos objetos. Se realiza un diagrama por cada acción y en él se indica la dirección del flujo de datos. Por ejemplo:

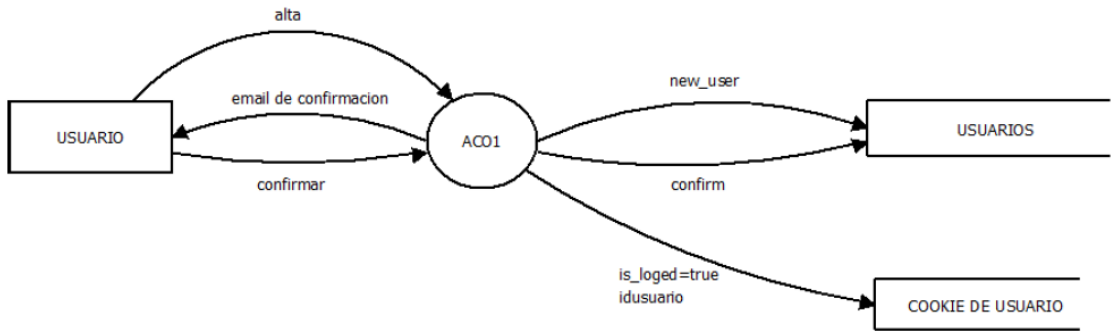

Ilustración 1. Diagrama de flujo de datos

En éste esquema los rectángulos representan los objetos del sistema, las cajas sin cerrar representan repositorios de datos y los círculos muestran el nombre del acontecimiento.

Postmortem: Ayuda a ahorrar tiempo de desarrollo porque es más rápido pensar antes de ponerse a programar y evita problemas futuros. Por otro lado debido quizás a la falta de experiencia hay ciertas erratas que ha habido que corregir, pero también es normal no poder anticipar todo.

3.3.2 Diagramas de secuencia

Descripción: Es una representación del orden en el que se producen las llamadas entre los componentes del sistema. En éste ya se representan los nombres de las funciones y clases (en mi caso es PHP) que intervienen. Es el diagrama de más bajo nivel que he realizado, con alguna excepción en las que he tenido que escribir pseudocódigo para

alguna funcionalidad especialmente complicada. Por ejemplo:

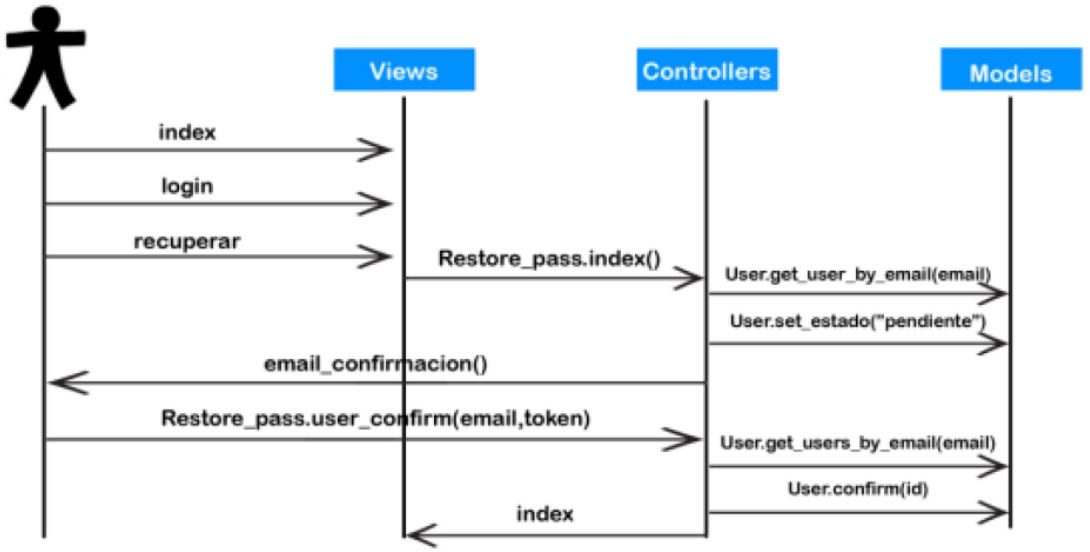

Ilustración 2. Diagrama de secuencia

Postmortem: Estos diagramas ahorran mucho trabajo. Las líneas de código por minuto que escribo se multiplican con ésta metodología. Son la chuleta de desarrollo por así decirlo.

3.3.2 Diagramas de estado

Descripción: Representación del estado de cualquier entidad que se crea, se modifica o se destruye durante la ejecución de la aplicación. Por ejemplo, la cookie de usuario.

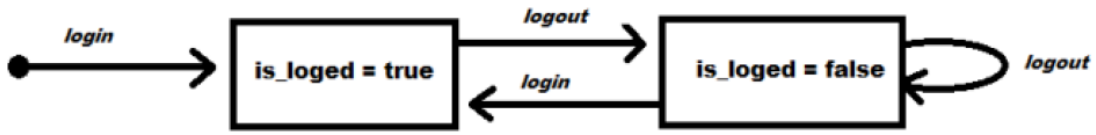

Ilustración 3. Diagrama de estado

Postmortem: Útil para las funcionalidades críticas de la aplicación. Por otro lado es una pérdida de tiempo si se confunde con los diagramas de flujo de datos. No se trata de adivinar la información que tenemos que presentarle al usuario, sino de los datos críticos para el funcionamiento correcto de la aplicación.

Al principio quise ser demasiado detallista en el estado de estos objetos, pero cualquier cambio en la interfaz provocaba cambios en todos los diagramas, así que finalmente opté por representar sólo los procesos que tienen un impacto fuerte y concreto en la aplicación.

3.4 Desarrollo del proyecto

Horas totales: 300 horas

Se ha desarrollado una página web y una aplicación para Android. Los lenguajes de programación utilizados han sido Java, PHP5, MYSQL, JavaScript, HTML5 y CSS3.

A continuación se expondrán los diferentes módulos que se han desarrollado, los problemas que se han presentado y las soluciones que se han propuesto.

3.4.1 Fase de implementación backend – frontend

Ésta fase incluye la implementación de todos los módulos en *PHP* y *JavaScript* de la aplicación y la interacción con la base de datos. Mis conocimientos al inicio del trabajo eran básicos y he invertido parte de mi tiempo en desarrollarlos.

Framework de desarrollo

Para la parte de servidor se ha utilizado el *framework CodeIgniter*.

CodeIgniter es un *framework* para aplicaciones web de código abierto para crear sitios web dinámicos con *PHP*. Su objetivo es permitir que los desarrolladores puedan realizar proyectos mucho más rápido que creando toda la estructura desde cero, brindando un conjunto de bibliotecas para tareas comunes, así como una interfaz simple y una estructura lógica para acceder esas bibliotecas.

También hay que destacar que *CodeIgniter* es más rápido que muchos otros entornos. Incluso en una discusión sobre entornos de desarrollo con *PHP*, Rasmus Lerdorf, el creador de *PHP*, expresó que le gustaba *CodeIgniter* «porque es rápido, ligero y parece poco un entorno».

El patrón de desarrollo que utiliza *CodeIgniter* es el Modelo-Vista-Controlador (*MVC*).

MVC es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello *MVC* propone la construcción de tres componentes distintos que son el **modelo**, la **vista** y el **controlador**, es decir, por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario. Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento

Además ofrece multitud de clases de ayuda que facilitan mucho el trabajo de implementación. Las clases que se utilizan más a menudo son:

- FormValidation, que incluye multitud de funciones para comprobación de campos de formularios
- Upload, para la subida de archivos al servidor

- Session, que tiene permite mantener el estado de usuario y seguir la actividad del usuario mientras navega la página.
- Email, para el envío de emails. Simplifica mucho la configuración y uso.

La base de datos:

Codeigniter utiliza un patrón personalizado de *Active Record*. Éste patrón permite realizar todo tipo de operaciones sobre la base de datos utilizando el mínimo código posible.

Además permite desarrollar aplicaciones independientes de la base de datos, porque la sintaxis es generada en función del adaptador de la base de datos utilizada, es decir, si desarrollamos la aplicación con un adaptador *MYSQL*, pero si en el futuro es necesario migrar a otro motor de bases de datos, no es necesario reescribir todas las consultas.

Y, por último, permite generar consultas más seguras, porque todos los valores son automáticamente higienizados por el sistema salvo que se le diga lo contrario, evitando de ésta manera inyecciones de *SQL* y *Cross Site Scripting*.

```
public function cargar_tickets_idmc($menusconfig)
{
    $array = array(
        'ticketmenu.idmenuconfig',
        'ticket.*'
        );
    $where_in = $menusconfig;
    return $this->db
        ->select($array)
        ->from('ticketmenu')
        ->join('ticket','ticketmenu.idticket=ticket.idticket')
        ->where_in('ticketmenu.idmenuconfig',$where_in)
        ->get()
        ->result();
}
```

Ilustración 4. Query con active record

Como se observa en la imagen las consultas realizadas con active record son mucho más legibles que escritas en una sentencia *SQL*, lo que reduce el tiempo dedicado a la depuración de las mismas.

3.4.1.1 Sistema de tickets y abonos

Los tickets y los abonos habitualmente sustituyen al dinero en los comedores por motivos como ofrecer descuentos por cliente habitual o por pago por adelantado de pensión completa o comidas como sucede en las residencias universitarias.

Desde el primer día tenía claro que éste sistema es la clave para que la aplicación sea un éxito o no. Si el proceso de venta y canje de tickets es complicado nadie querrá utilizarla por muy buenos resultados que en teoría pueda tener su aplicación.

Así, realicé un pequeño "*brainstorming*" de los pros y contras de unas y otras opciones. Las principales opciones que consideré, dando por sentado que el comedor cuenta con al menos un ordenador, son:

- Sistema de lector de tarjetas magnéticas o códigos de barras: Es un sistema probado y comprobado, pero los costes lo hacen inaccesible para ésta aplicación.
- Sistema lector *NFC* para móviles preparados con ésta tecnología o para un llavero "*token*". El lector *NFC* es más barato que un sistema *TPV*, unos 50€, y además ocupa poco espacio, es una solución moderna y que sin lugar a dudas veremos en muchas tiendas en el presente y en el futuro con la democratización del monedero electrónico. El principal problema es que la mayoría de las personas no tienen un Smartphone con *NFC*, y un llavero "*token"* por persona es un dispendio inasumible para una pequeña empresa. Si estuviera más extendida ésta solución desde luego me decantaría por ella.
- Sincronización vía internet entre móvil y caja del comedor. Se trata de una aplicación que emula el funcionamiento del sistema *NFC*, pero utilizando internet para transmitir la orden. Tiene el inconveniente que tanto comedor como usuario necesitan estar conectados a internet en el momento de la transacción. Por otro lado los gastos son nulos, sólo los de desarrollo de la aplicación, y la mayoría de las personas disponen de un Smartphone.

Finalmente opté por el último sistema. Su modo de empleo es el siguiente:

1. El cliente del comedor compra un abono (para comidas individuales no se aplica) en el comedor o en la página web. El abono puede utilizarse tanto para reservar menús, como para pagar comidas en el mismo momento sin reserva. El sistema se ha dejado así para mayor flexibilidad por parte de los comedores y que puedan ser ellos los que decidan si utilizar el sistema de reservas o no.
2. Si el comedor quiere que el cliente reserve con antelación, éste debe registrarse en la página web o desde la aplicación móvil y canjear el abono. Al cliente se le asignará un código (bautizado como *Nuki*), que llevará cuenta de la cantidad de tickets restantes (un solo *Nuki* para todos los tickets, por comodidad).
3. El cliente reservará desde la página web o desde el móvil los platos que desee.

4. Para “pagar” la comida el cliente desde la aplicación móvil sólo tendrá que apretar un botón y al comedor le aparecerán en pantalla los platos reservados y confirmará la transacción.

Para solucionar el caso en el que un cliente no tenga móvil el sistema permite al comedor la introducción manual de un código que se asigna a cada cliente cuando hace una compra en un comedor. El código se regenera tras cada uso y por tanto tiene el inconveniente de que el cliente debe apuntar el código y llevarlo encima. Para éstos casos he diseñado unas tarjetas que el vendedor del ticket puede entregar al cliente llegada la situación.

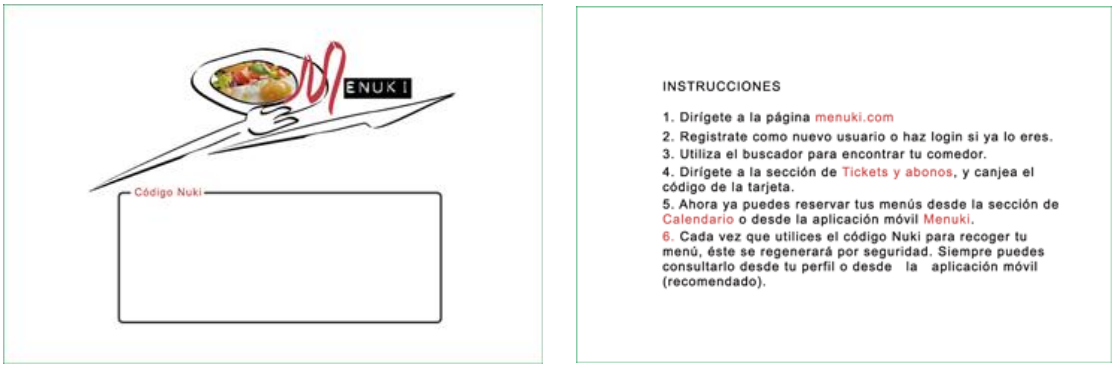

Ilustración 5. Tarjeta para códigos

Para el caso de no tener conexión a internet durante un periodo de tiempo:

- En el caso del comedor, la aplicación web utiliza *IndexedDB*, que es una tecnología de bases de datos añadida recientemente a *HTML5*, con la que se puede guardar información en el lado cliente y actualizarla cuando exista conexión a internet. De éste modo el navegador actúa como si fuera una aplicación nativa y permite una mayor flexibilidad y robustez.
- En el caso del cliente móvil, la información de los tickets del cliente se guarda en el teléfono móvil utilizando *Shared Preferences*, que son parejas clave valor que se almacenan encriptadas. Si el usuario ha tenido internet recientemente puede consultar su código en el móvil y transmitírselo al comedor para que lo introduzca manualmente como haría en el caso de las tarjetas descrito antes.

La implementación web de éste sistema se divide en varios apartados.

3.4.1.2 Creación de tickets y abonos

Descripción: Por una parte está la creación de un ticket, y por otra la creación de abonos de esos tickets. Por ej. Estaría el ticket de comidas, y el abono de 30 comidas que puede no valer lo mismo que 30 tickets de comidas.

Postmortem: Como en todos los formularios, se valida que la información es correcta y en éste caso que la operación la realiza alguien con permiso para realizarla. Una vez validado se introduce en la base de datos. Ahora el comedor puede generar abonos.

3.4.1.3 Generación de un PDF con varios abonos

Descripción: Para las ocasiones que un comedor reparta abonos de tickets, como por ejemplo, al inicio de un trimestre en una residencia, creí conveniente desarrollar un sistema que automatizara el proceso. El comedor crea un abono de los tickets que crea necesarios y después pinchando en un botón introduce el número de códigos que quiera generar y la aplicación le devuelve un PDF con el número de abonos formateados para que los recorte con una guillotina.

Postmortem: El proceso es el siguiente. El cliente web manda una petición vía AJAX al servidor con el abono y la cantidad que desea. En el servidor, tras generar los códigos y guardarlos en la base de datos, se genera un PDF utilizando la clase *FPDF* y se guarda en una carpeta del sitio web. Tras finalizar, se devuelve al cliente la URL del documento, que mediante la función de JavaScript *Windows.location* se abre en una ventana nueva.

3.4.1.4 Canjear un código y seguridad antifraude (Cliente)

Descripción: El cliente introduce en un formulario el código que le han entregado en el comedor y éste se añade a su cuenta. Para evitar fraudes, el sitio implementa un sistema de seguridad que si detecta 5 intentos en menos de un minuto bloquea los intentos de canjear al usuario; primero por 30 segundos, luego por un minuto y así exponencialmente.

Postmortem: Para implementar el sistema de seguridad hay dos opciones:

- Una muy segura pero poco eficiente, que es guardar la información de los intentos de canje en la base de datos. Esto puede provocar una avalancha de peticiones al servidor de BBDD.
- Otra menos segura, pero sin llamadas a la base de datos, que es guardar en la sesión de usuario y en la cookie los intentos realizados. Para saltarse ésta seguridad haría falta un script que cerrase la sesión del usuario y borrase la cookie entre cada intento, lo que ya supone una penalización de tiempo en sí.

  En ésta ocasión, siendo práctico, escogí la segunda. Pero la solución ideal pasaría por comprobar la fecha de creación de la sesión de usuario y si es muy reciente utilizar la opción de la base de datos, si no, utilizar la de la sesión. Esto queda como tarea futura.

Después de éstas comprobaciones de seguridad, si existe el código, se marca como usado y se añade a la cartera del usuario. Se redirige al usuario a la página de perfil donde puede comprobar el código y la cantidad de tickets de cada tipo asignados al mismo.

Ilustración 6. Tickets de usuario

3.4.1.5 Integración de la pasarela de pagos PayPal

Descripción: El proceso de venta de tickets en la web, está pensado para que los comedores puedan vender tickets online si lo desean. El usuario para esto debe rellenar un formulario con sus datos y se ofrece la opción de pago con PayPal.

La implementación está terminada, pero no será visible hasta que la empresa esté funcionando como tal. A continuación hago un breve resumen de en qué consiste integrar PayPal en un sitio web.

PayPal pone a disposición de los desarrolladores varias opciones para integrar PayPal en un sitio web dependiendo del tipo de sitio y de las necesidades que tengamos.

En éste caso se necesita una solución personalizada, porque los productos que tenemos varían en el tiempo. Para esto necesitamos generar desde nuestro sitio web botones encriptados autogenerados. Por desgracia desde nuestro país no es posible utilizar la API REST de PayPal, sólo está disponible para EEUU, así que se utiliza la API clásica.

El proceso en esencia es simple:

1. El servidor envía información del producto a PayPal junto con su identificación de cliente y PayPal responde con el código del botón encriptado.
2. Se incluye el botón encriptado de PayPal en la página, que redirigirá al cliente a PayPal donde puede utilizar varios métodos de pago o hacer Login y pagar con su cuenta si dispone de una.
3. Tras confirmar, PayPal redirige a una página de confirmación (en nuestra web), donde el cliente debe revisar el pago por última vez y procede a pagar.

Detrás de éste proceso hay toda una coreografía de mensajes entre la página web y PayPal. Los mensajes van cifrados mediante SSL de 168 bits (nivel más alto disponible comercialmente), para lo cual es necesario instalar en el servidor el consiguiente certificado SSL, proporcionar la información del mismo a PayPal y guardar en nuestro servidor un fichero cifrado, creado por PayPal, que es necesario para la autentificación en la comunicación.

Para utilizar la API de PayPal lo más acertado es buscar bibliotecas que gestionen la comunicación con cURL para evitar en lo más posible cometer errores.

Para probar el funcionamiento, PayPal dispone de un sitio de pruebas que es una copia del original, en el que se crean clientes ficticios y se utilizan datos y dinero ficticio.

3.4.1.6 Calendario de reservas (Usuario)

Descripción: El calendario de reservas muestra al cliente el menú disponible en el comedor. El usuario puede elegir el día para el que puede reservar y consultar las reservas que haya realizado. El usuario puede entonces seleccionar los platos y guardarlos, siempre que disponga de un ticket para ese plato.

Postmortem: La página web genera mediante JavaScript un calendario en el que se muestran los días para los que el usuario ya ha reservado. Éste script genera llamadas AJAX en segundo plano para cargar los menús a medida que el usuario navega por el calendario.

Para cada día se muestran los menús que haya disponibles (Comida, Cena…) y los platos. El envío de la selección del cliente se hace mediante AJAX también, y se permite modificar una selección previa solicitando una confirmación para ello.

3.4.1.7 Calendario de reservas (Admin)

Descripción: El administrador puede seleccionar un conjunto de platos para crear un menú, el ticket con el que se pagará y la fecha y hora máxima de reserva. El menú se puede configurar de manera que se repita en el tiempo y en el intervalo que considere más interesante.

Esta utilidad es muy cómoda para los administradores. Pueden crear una sola vez los menús y no volver a preocuparse de ellos hasta que decidan no repetirlos más, aunque todos sabemos que seguirá habiendo paella los jueves hasta el día del juicio.

Por supuesto los menús pueden eliminarse o modificarse. Si son menús que se repiten periódicamente se ofrece la posibilidad de editar uno sólo o toda la serie.

Además el calendario muestra el número de clientes que ha reservado cada plato en particular, permitiendo así calcular la cantidad que deberá cocinar de manera más ajustada.

3.4.1.8 Buscador

El buscador permite realizar búsquedas de comedores por nombre, dirección, descripción, etc. La manera de implementarlo se ve condicionada al motor de base de datos que estemos utilizando.

Existen dos maneras de buscar cadenas de texto en una base de datos, mediante la consulta LIKE o mediante FULLTEXT.

La consulta LIKE es una comparación de *cadenas* básica, la base de datos tiene que cargar el campo que se desea y buscar la cadena en cuestión dentro de él. Es una forma poco eficiente de hacer búsquedas en volúmenes grandes de información.

La consulta FULLTEXT, se indexan los campos para todas las palabras claves, omitiendo palabras que no aporten valor como las conjunciones. Los índices son creados como campos BLOB (*Binary Large Object*), al igual que en los documentos

Excel y Word. Por supuesto, el uso de esto aumenta la carga de memoria del sistema. Lo bueno que tiene es que se pueden realizar operaciones muy potentes, como búsquedas difusas.

Las búsquedas más rápidas en la base de datos MYSQL se realizan mediante consultas FULLTEXT. El motor de bases de datos *InnoDB*, que es el que utilizo porque permite realizar relaciones de clave foránea, sin embargo no permite en su versión 5.5 realizar búsquedas FULLTEXT. Para la versión 5.6 de MYSQL ya estará disponible, pero el servidor actual no lo admite. Si quisiéramos utilizar a toda costa FULLTEXT para mejorar las consultas, sería necesario convertir las tablas a *MyISAM*, lo que implica eliminar la tabla y los datos y volver a crearla con el motor *MyISAM*.

Teniendo en cuenta que la búsqueda sólo se realiza sobre dos tablas, y que en ellas hay una entrada por cada comedor y que esto no supone demasiadas lecturas, las consultas LIKE nos ofrecen la mejor solución en nuestro caso particular. En éste caso no compensa la sobrecarga de tener un buscador potente.

3.4.1.9 Caja (Administrador)

Descripción: La caja es el lugar desde el que se procede a validar los tickets en el comedor. El usuario acude con su código y el sistema muestra la selección del menú asociada con dicho código. También permite registrar ventas individuales, para más tarde generar estadísticas de ventas. Además también puede generar códigos de abono en el momento.

Esta página utiliza WebDatabases para permitir el funcionamiento sin conexión a internet. Al cargarse la página, se almacena la información de los menús, tickets asociados y todas las reservas que los clientes hayan realizado para ese día.

La página actualiza la información en esa base de datos y realiza periódicamente *polling* para comprobar si hay conexión disponible y transferir la información al servidor.

La implementación de la sincronización con dispositivos móviles para la transferencia del código también se realiza mediante *polling*, es decir, haciendo consultas desde el cliente al servidor para saber si tiene información para él.

Postmortem: La idea de este módulo es que funciones lo más parecido a una aplicación nativa posible. Todas las acciones de consulta se producen utilizando AJAX y no se recarga la página en ningún caso.

La caja tiene tres funcionalidades:

- Registrar ventas individuales: el dependiente marca los platos que ha cogido el cliente y éstos quedan guardados en la base de datos. Para las ventas individuales se pueden utilizar también códigos, si así lo ha configurado el comedor.

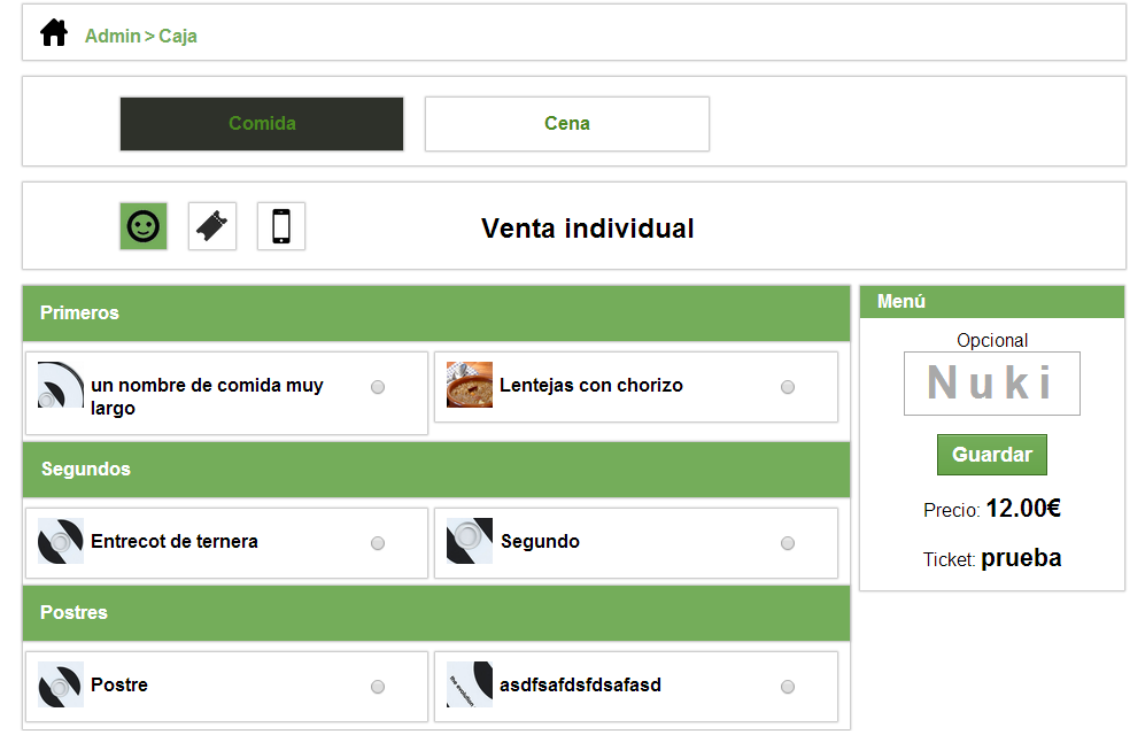

Ilustración 7. Caja - Venta individual

- Generar tickets: Se selecciona el tipo de ticket y la cantidad y se genera el código. El código generado se muestra en la página y éste será el que deba entregar al cliente para que lo canjee.

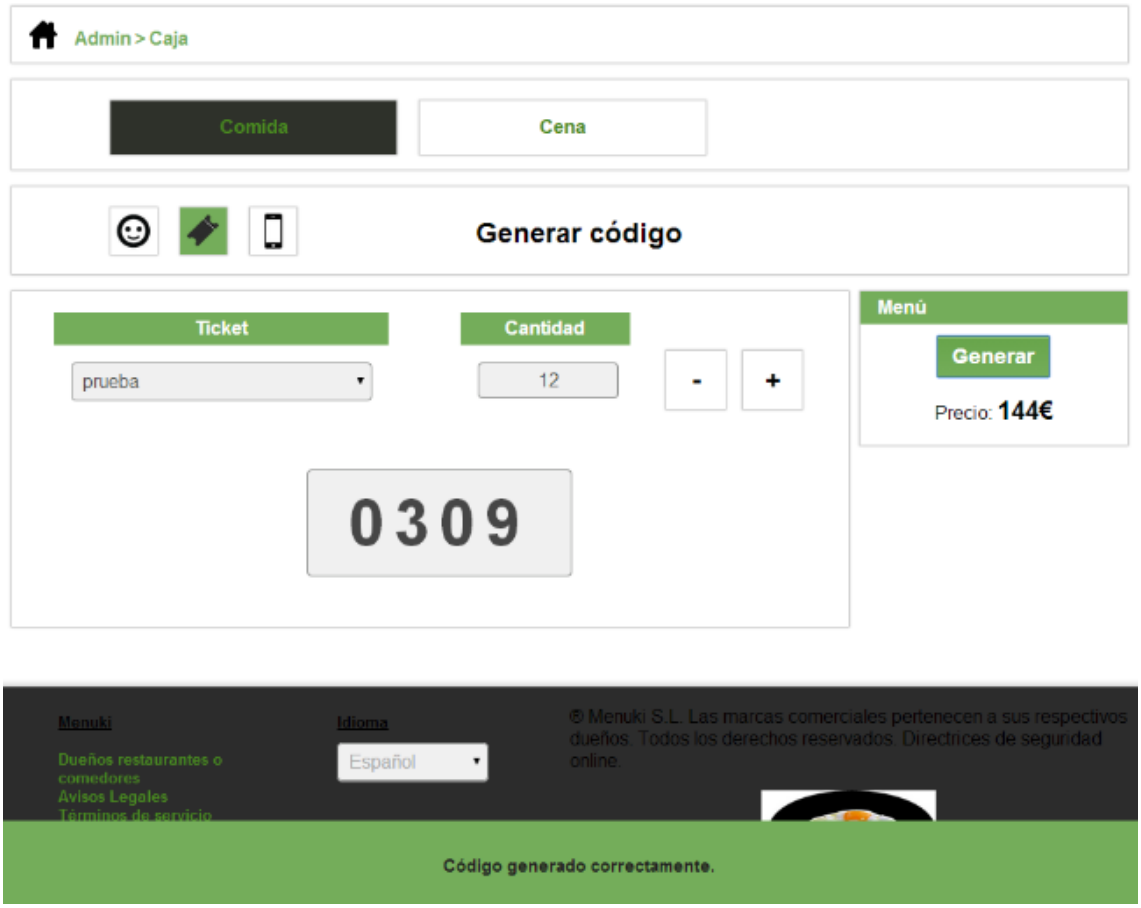

Ilustración 8. Caja. Generar código

- Canjear menú: el cliente, después de haber reservado con antelación el menú, canjeará el código de alguna de las maneras que expuse anteriormente (sincronizando desde el móvil o manualmente mostrando el código al dependiente). También se muestran el número de tickets restantes que están asociados al código del cliente.

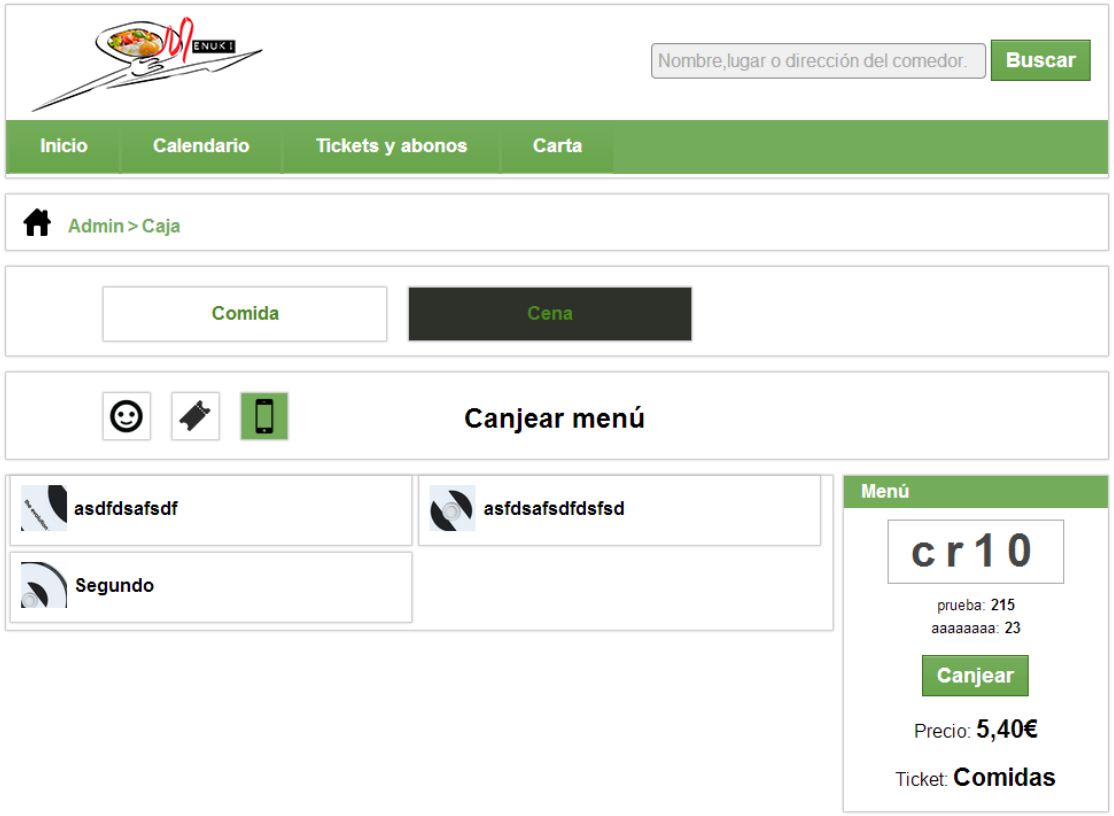

Ilustración 9. Caja. Canjear menú

3.4.1.10 Base de datos

Descripción: Se ha utilizado una base de datos relacional MySQL con el motor InnoDB. La base de datos utiliza relaciones de clave foránea, que permiten mantener la integridad de la información al realizar modificaciones sobre las tablas. La base de datos implementada tiene treinta tablas.

3.4.1.11 Implementación del alta de usuario

Descripción: Es todo el proceso de validación de formulario, introducir datos en la base de datos y confirmación de email.

Postmortem: Al final opté por la librería FormValidation de Codeigniter para validar los campos, a las que añadí funcionalidad propia de validación de DNI y NIE con expresiones regulares.

3.4.1.12 Implementación del alta de comedores

Descripción: Es un proceso de validar formularios, comprobar permisos de usuarios, subir imágenes e introducir datos en la base de datos.

Postmortem: Es una adaptación del trabajo hecho para el alta de usuario, aunque tiene características únicas como requerir estar logueado, por lo que la secuencia de acciones es un poco más compleja que en la anterior.

3.4.1.13 Implementación de la subida de fotos

Descripción: Permite al usuario recortar la foto de acuerdo a la forma en la que se va a mostrar y después subirla al servidor.

Postmortem: Utilizando JavaScript se toman las coordenadas por donde el usuario quiere recortar la imagen. Tras comprobar que los parámetros de formato de imagen y tamaño se cumplen, se recortan las imágenes utilizando la extensión de PHP ImageMagick que permite crear y modificar imágenes.

Ilustración 10. Recorte de imágenes

3.4.2 Fase de diseño e implementación de la interfaz web

Fase de maquetación y diseño de la web. Los lenguajes utilizados han sido HTML5 y CSS3 y los programas de diseño Adobe Photoshop y Adobe Ilustrator. Para la realización de esta parte me serví de multitud de recursos: libros, cursos online, tutoriales, vídeos.

Para el diseño de una página web lo primero que se debe hacer es hacer una lista de todas las funcionalidades que la página ofrecerá. Luego hay que agrupar éstas

funcionalidades por similitud entre ellas y de ésta manera ya tendremos la web dividida en páginas lógicamente agrupadas.

Una vez agrupadas hay que valorar la importancia de cada funcionalidad para el usuario final, de este modo a las funcionalidades más importantes les daremos una mayor visibilidad en la página ocupando sitios más altos y a la izquierda de la ventana.

Ilustración 11. Cómo leemos las páginas web

El método de posicionar la información importante arriba a la izquierda está basada en recientes estudios que han demostrado cómo buscamos la información en las páginas web. En la anterior imagen las zonas de calor muestran los puntos en los que el usuario del experimento detuvo más tiempo sus ojos. El primero en probar los mapas de calor fue el científico francés Louis Emile Javal en 1879, y se ha retomado recientemente.

3.4.2.1 Diseño de la cabecera

Descripción: La cabecera incluye:

- Barra de navegación de la aplicación: Incluye un logo con enlace a index, los botones de login y registro si el usuario no está logeado o los de perfil y logout.
- Cabecera de búsqueda: Esta cabecera puede ser personalizada por cada comedor para poner su imagen corporativa e incluye un cuadro de búsqueda.
- Barra de navegación del comedor: Cada comedor dispone de varias páginas a las que se puede acceder siguiendo los enlaces de esta barra de navegación.

Postmortem: Buscaba una solución que recordara a una página de blogs, porque lo que busco hacer es una red de micro blogs de comedores. Me inspiré en las páginas de WeBlog que es una red de blogs que incluye a Xataka, Genbeta, etc.…

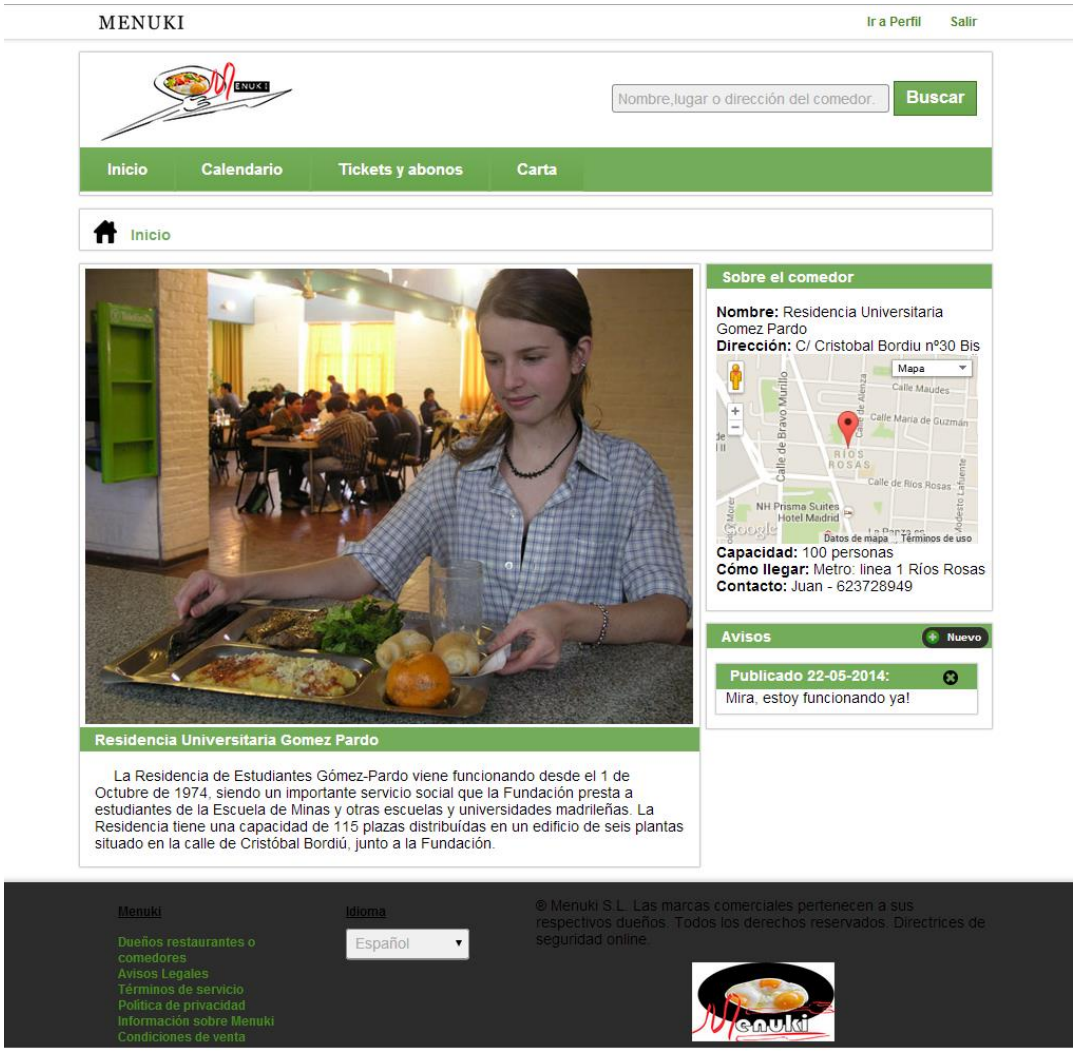

Ilustración 12. Página de inicio del comedor

3.4.2.2 Diseño del footer

Descripción: El footer contiene enlaces a declaraciones de privacidad, contacto, acceso para comedores. Es un bloque que tiene una funcionalidad limitada, y sólo incluye enlaces, un logotipo y un selector de idioma.

Postmortem: No es el elemento más importante de la página, pero es importante colocar todo bien ordenado, que resalte sobre el resto y que aparezca siempre pegado al fondo de la página, haya o no contenido que ocupe toda la página.

3.4.2.3 Diseño de la página de calendario (cliente)

Descripción: Es una página en la que se visualiza el menú de cada día y en la que se pueden seleccionar los platos. Permite navegar de un día a otro, bien seleccionando el día en un calendario o mediante botones de adelante y atrás.

Postmortem: Esta página tiene la interacción clave de la página y era importante que fuera simple y clara.

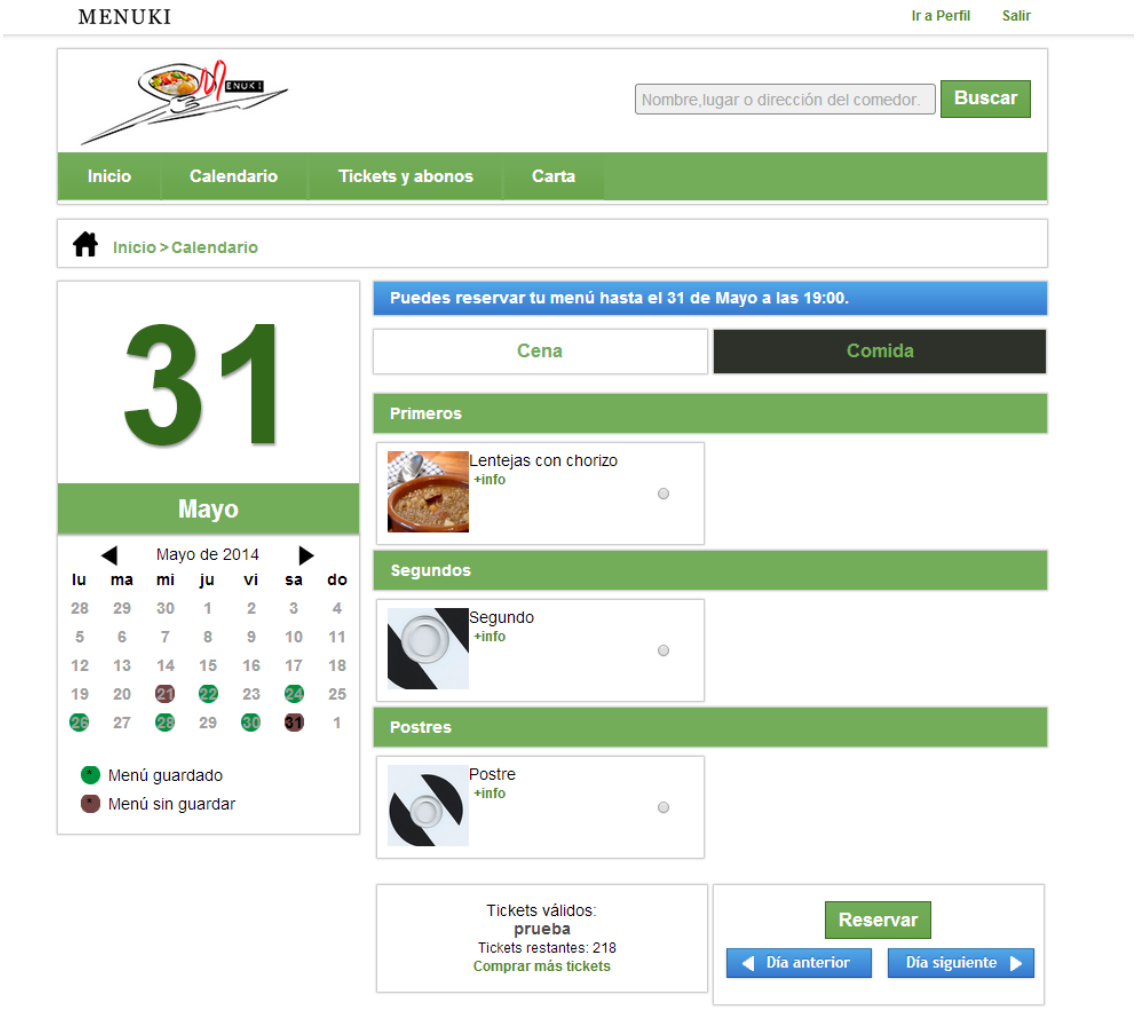

Ilustración 13. Página de calendario de usuario

3.4.2.4 Diseño de la página de calendario (administrador)

Descripción: Es un formulario para dar de alta menús. La interfaz es clara pero a la vez permite realizar multitud de configuraciones.

Postmortem: Tomé prestadas un par de ideas de google calendar. Debía de ser como una navaja suiza, ser simple pero a la vez tener muchas funcionalidades que estuvieran ahí cuando las necesitaras.

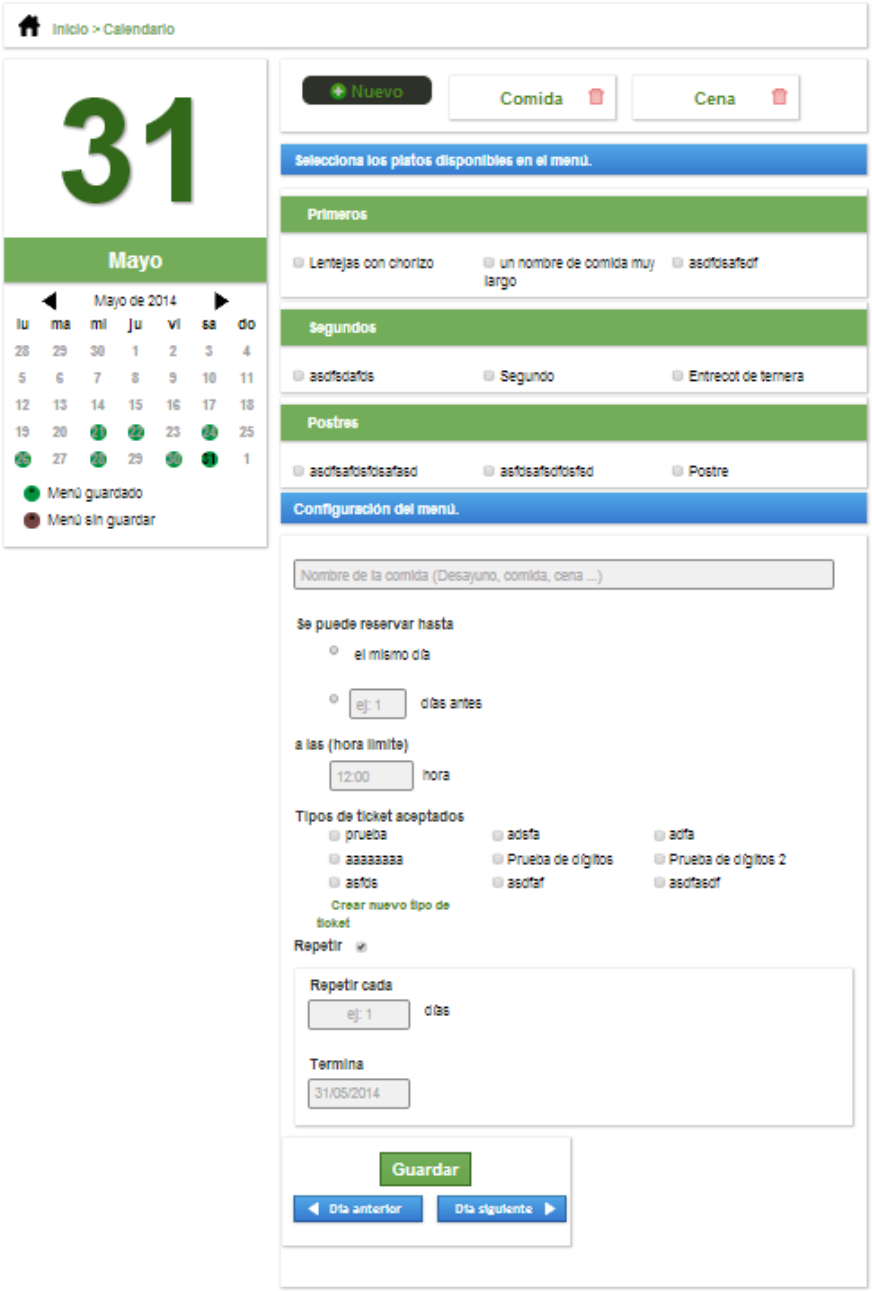

Ilustración 14. Página de calendario administrador

3.4.2.5 Diseño de la página de venta

Descripción: Interfaz sencilla que muestra una lista con los tickets que han puesto a la venta los comedores. En el fondo es un formulario y funciona como una tienda para los comedores.

Postmortem: Consideré importante resaltar el precio por menú y la descripción del producto.

3.4.2.6 Diseño de las páginas de proceso de venta

Descripción: Formulario de facturación, selección de método de pago y confirmación.

Postmortem: Tuve problemas con la maquetación de algunos campos en los formularios y me retrasó bastante.

3.4.2.7 Diseño de las páginas de proceso de alta de usuario

Descripción: Formulario de alta de usuario. Incluye tips de ayuda para los campos que puedan generar dudas por el formato de entrada.

Postmortem: En éste caso los problemas con los formulario ya estaban resueltos por el punto anterior, pero los tips de ayuda que se muestran con javascript al moverse el usuario de un campo a otro llevaron tiempo. Fue necesario consultar tutoriales online y ayuda en páginas como stackoverflow.

3.4.2.8 Diseño de la página de inicio

Descripción: Página de inicio de la aplicación. Explica cómo funciona la página y cómo darse de alta y buscar comedores.

Postmortem: Página de aterrizaje para los usuarios que entren por primera vez, la información debe de ser directa. Tras estudiar varios artículos que discutían el tema al final opté por imágenes grandes que llamen la atención y provoquen una reacción.

Ilustración 15. Landing page

3.4.2.9 Diseño de la página de inicio para comedores

Descripción: Es una página en la que los comedores pueden presentar información de contacto, cómo llegar, el menú del día, imágenes del comedor y avisos que publique el administrador del comedor.

Postmortem: Tuve que hacer una maquetación nueva para casi todos los elementos de la página, por lo que tardé bastante tiempo en dejarla cómo la había diseñado.

3.4.2.10 Diseño del alta de un comedor

Descripción: Formulario de alta de comedor.

Postmortem: A estas alturas las maquetación y estilos de los formularios ya estaban definidos por lo que me llevó menos tiempo. Hubo que cambiar algún mensaje de error y pocas cosas más.

3.4.2.11 Diseño de las páginas de perfil de usuario

Descripción: Una página desde la que el usuario puede acceder a sus datos personales e historial. Incluye las páginas en las que se muestra la información histórica en forma de tablas ordenadas.

Postmortem: Me inspiré en la página de perfil de usuario de PCcomponentes. Tiene un estilo por fichas muy agradable y sencillo. Me costó tiempo encontrar los iconos adecuados para la página y que además fueran gratis para su uso comercial, para ello utilicé la página de iconfinder.

3.4.2.12 Diseño de las páginas de perfil de administrador

Descripción: Es una página similar a la de usuario, pero incluye gráficas para visualizar la evolución del negocio.

Postmortem: En éste caso aunque la maquetación y los estilos estaban desarrollados integré un plugin javascript para generar gráficas, entre las diferentes opciones que había finalmente me decanté por Chart-js. La librería es sencilla de utilizar.

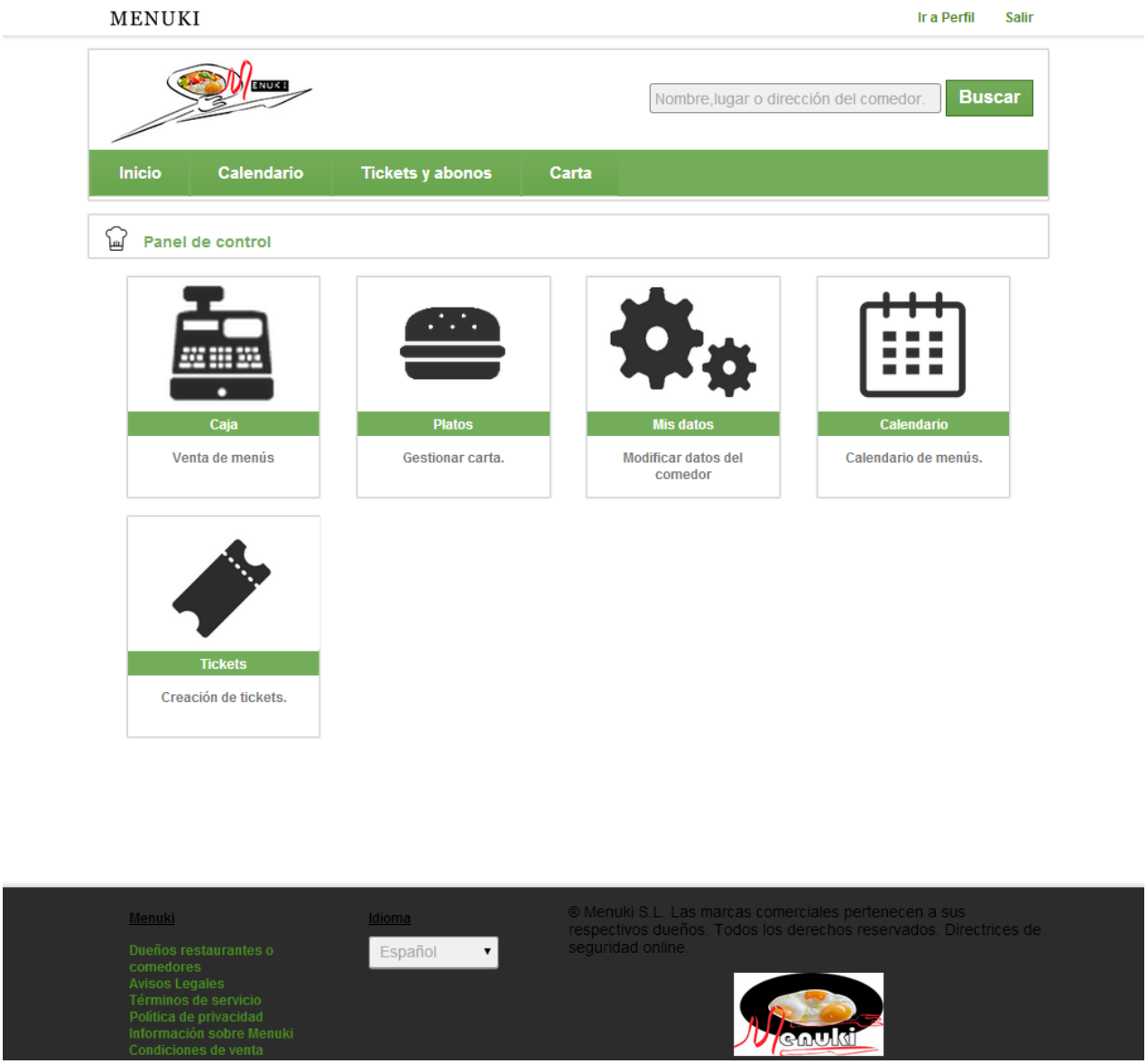

Ilustración 16.Perfil de control de administrador

3.4.2.13 Diseño de la página de login

Descripción: Formulario de login.

Postmortem: Otro formulario más. Sólo hubo que realizar retoques de maquetación y de errores.

3.4.3 Desarrollo de la aplicación móvil: Android

Para complementar la aplicación web se está desarrollando una aplicación móvil, que permita a los usuarios consultar los menús de los comedores, reservar y gestionar sus tickets.

Se ha elegido Android para realizar la primera aplicación móvil debido a que es el sistema operativo móvil más extendido y para el que más aplicaciones existen desarrolladas. Debido a esto, se pueden encontrar muchas guías y tutoriales para crear todo tipo de aplicaciones.

El entorno para desarrollar la aplicación ha sido eclipse con el ADT Bundle que permite desarrollar aplicaciones para Android e incluye un emulador para hacer pruebas, aunque se ha utilizado mi propio teléfono móvil para las mismas.

3.4.3.1 Estructura de la aplicación

Se divide en:

- Login y registro: Permite al usuario crear una nueva cuenta o acceder con sus credenciales si ya dispone de una.
- Tickets de usuario: Muestra los códigos de los tickets y la cantidad restante de cada tipo de ticket. Permite canjear los códigos haciendo al tocar con el dedo.
- Búsqueda de comedor: Un buscador que muestra una lista de los comedores encontrados para determinada búsqueda.
- Reserva de menús: Muestra los menús disponibles por día, y para cada uno, muestra la lista de plato. Permite reservar la comida con antelación, además de consultar el menú del día.

Android divide las aplicaciones en actividades (*Activity*). Las actividades integran una funcionalidad de la aplicación que el usuario puede hacer en un determinado momento. Cada actividad genera una nueva ventana e interactúa con el usuario, generando eventos cuando el usuario realiza alguna acción como pulsar un botón o deslizar el dedo.

3.4.3.2 Comunicación con el servidor

3.4.3.3 Navegación

3.4.3.4 Otras funcionalidades: Buscador por mapa. Google Services. Google Maps

Para complementar la búsqueda de comedores, se ha integrado la aplicación con Google Maps. Así, conociendo la geolocalización del usuario se pueden mostrar los comedores que hay en los alrededores, y acceder directamente a ellos.

Para acceder a Google Maps, se necesita una clave API que google proporciona a los desarrolladores interesados.

El proceso para integrar los comedores en el mapa es el siguiente:

1. Se solicita el servicio de mapas al sistema operativo, así como la posición del móvil. Se puede conocer si el usuario tiene el GPS activo o no, para un posicionamiento más fino, pero en éste caso nos conformamos con el que esté disponible. Una vez creado el objeto mapa, puedes interaccionar con él, como con cualquier objeto Java.
2. Se extraen latitud y longitud de las esquinas del mapa y se envían en segundo plano al servidor.
3. En el servidor se buscan los comedores cuya geolocalización esté entre las coordenadas enviadas y se envía la información de vuelta al terminal como un texto en JSON. La latitud y longitud de los comedores se había calculado en el proceso de alta.
4. Por cada comedor se coloca un marcador en el mapa.

El proceso en sí es bastante sencillo, pero los mapas consumen mucha memoria y es muy fácil tener problemas de rendimiento en los móviles de un solo núcleo. Por eso es importante crear y destruir correctamente los objetos cuando las actividades pierden el foco de los usuarios. En el caso de los mapas, sumamos que manejamos un objeto que es muy caro, en términos de recursos, de crear, y por eso hay que guardarlo en cache para evitar bloquear la aplicación muy a menudo.

4. RESULTADOS Y CONCLUSIONES

El trabajo fin de carrera ha sido una gran oportunidad, que creo no haber desaprovechado, para probar los conocimientos adquiridos durante la carrera. Al inicio del trabajo, la empresa era sólo una idea y ahora es algo tangible, tanto como puede llegar a ser el software.

Los objetivos propuestos al inicio del trabajo y la fecha de entrega del trabajo, han supuesto una presión extra y han aportado la motivación necesaria para seguir adelante con la idea hasta el final. Lo más importante es terminar.

5. ANEXOS

5.1 Diagrama de Gantt

6. BIBLIOGRAFÍA

6.1 Bibliografía básica

[1] P. J. Lynch and S. Horton (2009, Jan 15), *Web Style*. *(3rd Edition)* [Online] Available: http://www.webstyleguide.com/index.html
[2] D. Upton, *Codeigniter for rapid PHP application development*. PACKT publishing(2007)
[3] K. Tatroe, P. MacIntyre, R. Lerdorf, *Programming PHP* (3rd Edition) O'Reilly Media (February 22, 2013)
[4] D. Flanagan, *JavaScript: The Definitive Guide* (6th Edition) O'Reilly Media (May 13, 2011)
[5] C. Schmitt, K. Simpson, *HTML5 Cookbook* (2nd Edition) O'Reilly Cookbooks (2011, Nov 26)

6.2 Bibliografía específica de diseño web

[6] ruck(2013,Dic 9), *Landing Page Tutorials And Resources* [Online] Available: https://www.imgrind.com/21-awesome-landing-page-tutorials-and-resources/

[7] alrra (2013 Abr 1), *HTML5 Boilerplate documentation*. [Online] Available: https://github.com/h5bp/html5-boilerplate/blob/v4.2.0/doc/TOC.md

[8] Miseades (2013 Abr 16), *Las 30 (imprescindibles) tipografías por las que jamás serás criticado.* [Online] Available: http://misedades.wordpress.com/2013/04/16/las-30-imprescindibles-tipografias-por-las-que-jamas-seras-criticado/

[9] Diane (2013, Sep 10*), Estilo Metro. Tendencia Fresca en el Diseño Web.* [Online] Available: http://www.templatemonsterblog.es/2013/04/11/estilo-metro-tendencia-fresca-en-el-diseno-web/

[10] Google (2014), *Customizing Google Maps: Custom Markers*. [Online] Available: https://developers.google.com/maps/tutorials/customizing/custom-markers

[11] Petr Stanicek (2010), *Color Scheme Designer* (v3.51) [Online] Available: http://colorschemedesigner.com/

[12] Responsinator(2013), *Herramienta online para comprobar diseños responsive*. [Online] Available: http://www.responsinator.com/

[13] Mockingbird (2013), *Herramienta para realizar bocetos de webs*. [Online] Available: https://gomockingbird.com/mockingbird/

[14] CSS3 Button Generator (2013*), Herramienta online para generar botones*. [Online] Available: http://www.cssbuttongenerator.com/

6.3 Bibliografía específica de javascript

[15] B. A. Lupton (2011-1014), *jQuery Browserstate para HTML5*, [Online] Available: https://github.com/browserstate/history.js

[16] 42 contributors (2010-2014), *JavaScript reference*. [Online] Available: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference

[17] Webdevelopersnotes (2011), *10 ways to format time and date using JavaScript*. [Online] Available: http://www.webdevelopersnotes.com/tips/html/10_ways_to_format_time_and_date_using_javascript.php3

[18] Andrey (2012,Oct 12), *HTML5 Image uploader with Jcrop*. [Online] Available: http://www.script-tutorials.com/html5-image-uploader-with-jcrop/

6.4 Bibliografía específica de PHP

[19] M. Dole (2002, Nov) *Working with Dates*. [Online] Available: http://www.elated.com/articles/working-with-dates/

[20] M.A. Alvarez (2009, Nov 23*) Instalación y configuración de CodeIgniter*. [Online] Available: http://www.desarrolloweb.com/articulos/instalacion-configuracion-codeigniter.html

[21] FPDF(2014) *FPDF, clase PHP para generar PDF's*. [Online] Available: http://www.fpdf.org/?lang=es

[22] P.Ventura (2011, Mar 29) *Problemas codificación web. Configurar codificación UTF8. PHP, MYSQUL y HTML*. [Online] Available: https://www.pedroventura.com/php/problemas-codificacion-web-configurar-codificacion-utf8-php-mysql-y-html/

[23] U. Jiménez (2007, Agosto 13), *Generar PDF con PHP y MySQL*. [Online] Available: http://blog.unijimpe.net/generar-pdf-con-php-y-mysql/

[24] pagecolum.com (2014*), PHP Regular Expression Tester*. [Online] Available: http://www.pagecolumn.com/tool/pregtest.htm

[25] weblogtoolscollection.com (2014) *Regular Expression Tutorial.* [Online] Available: http://weblogtoolscollection.com/regex/regex.php

[26] S. Dzysyak (2013) *Geocoding con Google Maps api v3 PHP*. [Online] Available: http://erlycoder.com/45/php-server-side-geocoding-with-google-maps-api-v3

[27] israel965 (2013, Ene 20*) Login con permisos de usuarios en codeigniter*. [Online] Available: http://uno-de-piera.com/permisos-de-usuarios-en-codeigniter/

[28] codeigniter (2014*) Codeigniter's Activer Record Database Pattern*. [Online] Available: http://ellislab.com/codeigniter/user-guide/database/active_record.html

[29] devdocs (2014) *Documentación centralizada de multitud de tecnologías web*. [Online] Available: http://devdocs.io/

[30] JamesG (2011) *Cliente PHP SOAP para API de Paypa*l. [Online] Available: http://stackoverflow.com/questions/3105577/simple-php-soapclient-example-for-paypal-needed

[31] PayPal (2012, Oct*), Guía de integración de Pasarela integral*. [Online] Available: https://www.paypalobjects.com/webstatic/es_ES/developer/docs/pdf/pasarelaintegral_es.pdf

[32] prosoxi.com (2012, Mar), *Installing SSL using OpenSSL on a WAMP localhost.* [Online] Available: http://www.prosoxi.com/2012/03/14/installing-ssl-using-openssl-on-a-wamp-localhost/

[33] Paypal (2014), *Documentación API Paypal.* [Online] Available: https://developer.paypal.com/webapps/developer/docs/classic/paypal-payments-standard/integration-guide/Appx_websitestandard_htmlvariables/#id08A6HF00TZS

[33] codeigniter(2013), *Mejor .htaccess para codeigniter*. [Online] Available:

http://www.farinspace.com/codeigniter-htaccess-file/

[34] B. Guzel (2009, 25 Nov), *Top 20+ MySQL Best Practices*. [Online] Available:

http://code.tutsplus.com/tutorials/top-20-mysql-best-practices--net-7855

6.5 Bibliografía específica de HTML y CSS

[35] livetooles.uiparade.com (2014*) Herramienta Form builder*. [Online] Available: http://livetools.uiparade.com/form-builder.html#

[36] w3schools.com (2014) *CSS3 Animations*. [Online] Available: http://www.w3schools.com/css/css3_animations.asp

[37] cubic-bezier.com(2014) *CSS3 Animations, cubic-bezier*. [Online] Available: http://cubic-bezier.com/

[38] westciv.com (2014) *Herramientas CSS3*. [Online] Available: http://www.westciv.com/tools/gradients/

[39] MDN (2014) *CSS reference*. [Online] Available: https://developer.mozilla.org/en-US/docs/Web/CSS/Reference

6.6 Bibliografía específica de Android

[40] VIJAYAKUMAR M (2013) *Android lazy image loader example*. [Online] Available: http://iamvijayakumar.blogspot.com.es/2011/06/android-lazy-image-loader-example.html

[41] A. Alcalde (2014,Ene 16), *Cómo crear un adapter personalizado en Android*. [Online] Available: http://elbauldelprogramador.com/adapter-personalizado-en-android/

[42] Android-developers.blogspot.com.es (2009, Feb 24), *Android Layout Tricks*. [Online] Available: http://android-developers.blogspot.com.es/2009/02/android-layout-tricks-1.html

[43] rmonannurik.github.io (2014) *Android Asset Strudio (Herramientas).* [Online] Available: http://romannurik.github.io/AndroidAssetStudio/

[44] R. Tamada (2013, Ago 25*), Android working with Google Maps V2*. [Online] Available: http://www.androidhive.info/2013/08/android-working-with-google-maps-v2/

[45] R. Tamada (2013, Oct 6), *Android Tab Layout with Swipeable Views*. [Online] Available: http://www.androidhive.info/2013/10/android-tab-layout-with-swipeable-views-1/

[46] Google (2014) *Creating a Navigation Drawer*. [Online] Available: http://developer.android.com/training/implementing-navigation/nav-drawer.html

[47] nostra13 (2014*) Powerful and flexible library for loading, caching and displaying images on Android (Android-Universal-Image-Loader).* [Online] Available: https://github.com/nostra13/Android-Universal-Image-Loader

[48] skill-guru.com (2011) *Data Storage Mechanisms in Android*. [Online] Available: http://www.skill-guru.com/blog/2011/01/17/data-storage-mechanisms-in-android/

[49] Google (2014) *Draw-9 patch. Herramienta de creación de imágenes para distintas densidades de pantalla Android*. [Online] Available: http://developer.android.com/tools/help/draw9patch.html

[50] R. Tamada (2012, Feb 20). *Android Custom ListView with Image and Text*. [Online] Available: http://www.androidhive.info/2012/02/android-custom-listview-with-image-and-text/

[51] R. Tamada (2012, May 2), *How to connect Android with PHP, MySQL*. [Online] Available: http://www.androidhive.info/2012/05/how-to-connect-android-with-php-mysql/

[52] R.Tamada (2012, Jan 31) *Android Login and Registration with PHP, MySQL and SQLite*. [Online] Available: http://www.androidhive.info/2012/01/android-login-and-registration-with-php-mysql-and-sqlite/

[53] UNITiD (2014) *Android Design Patterns*. [Online] Available: http://www.androidpatterns.com/

[54] Google (2014) *Android Application Design*. [Online] Available: http://developer.android.com/design/get-started/ui-overview.html

[55] petrnohejl.github.io (2014*) Android cheatsheet for graphic designers.* [Online] Available: http://developer.android.com/design/get-started/ui-overview.html

[56] foro (2014) *Calling a web service fom Android*. [Online] Available: http://www.anddev.org/calling_a_web_service_from_android-t348.html

[57] binpress (2014) Clase PHP con para API de PayPal [Online] Available:

http://www.binpress.com/app/php-paypal-api-class/20

Este documento esta firmado por

	**Firmante**	CN=tfgm.fi.upm.es, OU=CCFI, O=Facultad de Informatica - UPM, C=ES
	Fecha/Hora	Fri Jun 06 23:49:07 CEST 2014
	Emisor del Certificado	EMAILADDRESS=camanager@fi.upm.es, CN=CA Facultad de Informatica, O=Facultad de Informatica - UPM, C=ES
	Numero de Serie	630
	Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)